Bellingham OS-9 Users Group

Gimix, CoCo, Atari, Mac
6809 - 68K OS-9 Level 1, 2, 3

Volume I No. 5 April 30,1990

OS-9 MEETINGS:

Meetings are held at 7:30 p.m., the second Thursday of each month in room 109 at Sehome High School

BENEFITS TO MEMBERS:

As a participating member of our new Bellingham OS9 Users Group you enjoy many benefits:

1. Newsletter
2. OS9 Bulletins
3. Public Domain Library
4. Technical help
5. Lectures and demonstrations
6. Periodic group purchases
7. Membership List
8. Access to GIMIX Level-III OS9

HELP WANTED!

Our group needs editorial volunteers. If you can contribute with information or helpful experiences of your own, please contact Rodger Alexander. The health of our newsletter depends on contributions made by many members of our group.

THIS ISSUE:

SUBSCRIPTION INFORMATION:

Newsletters are available free to those in attendance at the monthly meetings. If you would like to receive the newsletter in advance by mail a subscription rate of $3 for 6 monthly issues or $6 for 12 monthly issues is available.

Contact: Rodger Alexander
3404 Illinois Lane
Bellingham, WA 98226
(206) 734-5806
Delphi: "SALZARD

PLEASE NOTE: MAY MEETING IS THURSDAY THE 17th INSTEAD OF THE 10th

OTHER SOURCES:
When you buy a Color Computer from Radio Shack, a flyer is included with sub-scription infromation for "THE RAINBOW" Magazine. published by Falsoft. Inc. At one time there were three publications that were specifically written for the Color Computer. Besides "THE RAINBOW" there was also "HOT COCO" and the "COLOR COMPUTER NEWS". My favorite was "HOT COCO", and I have every issue. But when the format of the magazine started changing you could see the end coming. Now, only "THE RAINBOW" remains. although it has been shrinking from over 250 pages to 130.

68 MICRO JOURNAL
NOT TRUE! There are actually several magazines available that pertain to the Color Computer and/or OS9. THE 68 MICRO JOURNAL has been with us from the very beginning covering all 6809 - 68000 computers with emphasis on OS9. FLEX and other Operating Systems. "THE 68MICRO JOURNAL" has been oriented more towards the programer and com-mercial and business users rather than the adventure game player. "THE 68MICRO JOURNAL" was just too much "over our heads" (speak for yourself bub!), so few "CoCo Nuts" were even aware of "THE 68MICRO JOURNAL". The "JOURNAL" has now gone almost strictly to covering OSK (68000 version of OS9), which is again aimed at the professional and commercial applications. Owners of the new KLE's "MM1" and FHL's TC9 might find the "JOURNAL" very useful, but information on our favorite version of OS9 will be pretty rare.

COCO CLIPBOARD MAGAZINE
The "CLIPBOARD" put out it's first issue in October 1987 and in many ways has grown to resemble a smaller version of "THE RAINBOW". This is not a put down and the publisher has worked hard at filling the gaps that "the other" magazine misses. The "COCO CLIPBOARD" is published bi-monthly and includes reviews, programs, reader mail and tutorals. An Amateur Radio article is included in every issue. One thing different is advertizer's coupons included in every issue. Like "the other mag-azine" the "CLIPBOARD" also has "CLIPDISK", all of the programs printed in the magazine on disk, for $49.95 a year. The most recent issue of the "CLIPBOARD" was 49 pages in length. A year's subscription (6 issues) is $18. Credit cards accepted. Address: 3742 U.S. 20. Box 3, Fredonia, NY 14063 (716) 679-0126.

MOTD
Stands for Message Of The Day. "The International Newsletter of the OS-9 Users Group". The header reads "OS9 Macintosh! OS9 Clone! OS9 Amiga! OS9 Atari! OS9 CoCo! OS9 GIMIX!". The order of the different OS9 Systems is made very apparent with each succeeding computers printed in smaller and smaller fonts. CoCo appears to be next to last, but I found the CoCo featured on par with the others. And then I noticed that the President of the O Users Group is Kevin Darling, the Editor is Bill Brady, Bruce Isted is Vice President and Dale Puckett is Director-at-Large. Most of the reviews were for software originally written for the CoCo, although most of it (non graphics) would run on all OS9 systems. This magazine is obviously for the OS9 fans only. Software listings are not included but OS9 development articles and software reviews makeup the major contents of the 10-12 page bi-monthly newsletter. The OS9 Users Group also maintains an extensive Library (20 DoubleSided 40 track Disk) of OS9 programs and utilities. Subscription to the "MOTD" Newsletter comes with a $25 membership in the OS9 Users Group. Address is 1715 East Fowler Ave., Suite R237. Tampa. FL 33612. The OS9 Users Group Library of 20 Disk is $100.

COCO NOTES NEWSLETTER
This is a brand new endeavor. The June/July issue will be it's 1 Ye anniversary issue. The magazine is devoted to 50% OS9 and 50% RS-Dos. The newsletter is in a magazine format.

a.though several previous issues were available only on disk and the subscription is available in both printed format and disk format. The anniversary disk will contain 27 to 30 articles, (more than the printed format). The subscription rate is $12/year and the address is CoCo Notes Newsletter, P.O. Box 45434, Tacoma, WA 98445

HEY!!! LETS GET THIS GUY TO TALK AT ONE OF THE MEETINGS

RADIO SHACK COCO SALE

No this is not a paid advertizement, BUT.....

300 baud Modem Pak	$9.95
Deluxe Color Mouse	$29.95
CM-8 Analog Color Monitor (works with Amiga and Atari ST)	$199.95
Color Computer-3	$129.95
CoCo Software 32% to 67% Off	

[illegible] CAN YOU DO WITH A 300 BAUD MODEM?

[illegible]acked my Modem pak this past weekend and converted it into a Midi pak (well, half a Midi pak -- it'll xmit Midi but I'm still waiting on a couple parts to finish the receiving section). I verified that it worked by PEEKing and POKEing the pak from RS Basic. It seems to work fine.

I still have to:

- Hack the /MIDI driver that comes with UMuse (disassemble enough to know what's going on, patch the driver, etc). Disassemble?? Did I say that???
- Hack the /SIO driver (similar to the /T1 driver). (But this was for the serial port, if memory serves.)
- Hack the /T2 driver (copy to /MIDI, XMode it, etc).

------John Sheer-----

CM-8 ON AN AMIGA?

[illegible]t quote me, but I think the CM-8 pinout is:

Gnd Red Blu Snd Ver

(Hor=horiz, Ver=vertical, Snd=sound
Gnd Grn Hor Pia
Pia=internal pia pin - ignore)

The amiga monitor is:

Gnd Gnd Red Grn Blu
n/c Sync n/c n/c

To use with an Amiga monitor (which expects composite opposite sync) you have to use a 74LS02 NOR gate. Take the coco Horizontal sync and Vertical sync into pins 2,3.. the output at pin 1 of the chip will be the composite Sync required.

I use an Amiga monitor on my CoCo, as the CM-8 is on the KMA and the Amiga is in the closet <grin>.

-----Kevin Darling-----

MODIFY YOUR C COMPILER

This document is intended to assist the new C Developer with some of the things not covered in the manuals. First of all, READ THIS WHOLE DOCUMENT BEFORE YOU DO ANYTHING!!! It could save you some time.

The first part covers installing the Development Pack. Later it tells how to accommodate what it gives you with the C Compiler.

Next it tells how to move Microware's OS9 C Compiler to a single disk. This can be a hard drive or a large (3.5 720K) floppy. This is followed by a discussion of various alternatives to using cc1.

This file refers to devices /d0 and /dd. /d0 is assumed to be the device in which the original floppy distribution disks are going to be read from. /dd is assumed to be the target disk to install the C Compiler on.

This procedure also requires the save command. It is on the Development Pack. If you don't have the Development Pack

and merely need to patch the C modules. you will need to obtain a replacement save command. Kevin Darling has written one (posted on CompuServe and Delphi) which is a DECB program which creates the binary for you.

This procedure involves the creation of several directories which you may or may not have.

INSTALLING THE DEVELOPMENT PAK

The Development Pack comes with several files which aid in development with the Microware C Compiler. First of all, if you are using a hard drive, I recommend copying all of the additional commands. Put *A COPY* of Side A of the floppy in device /d0 and enter:

```
OS9: chd /d0/cmds
OS9: dsave /d0 /dd/cmds ! shell
OS9: chd /d0/sys
OS9:   merge   errmsg   /dd/sys/errmsg
     >/dd/sys/errmsg.new
OS9: del errmsg
OS9:  merge  helpmsg  /dd/sys/helpmsg
     >/dd/sys/helpmsg.new
OS9: del helpmsg
OS9: dsave /d0 /dd/sys ! shell
OS9: chd /dd/sys
OS9: del errmsg
OS9: rename errmsg.new errmsg
OS9: del helpmsg
OS9: rename helpmsg.new helpmsg
```

Take out that floppy and put in A COPY of Side B and enter:

```
OS9: chd /d0/defs
OS9: dsave /d0 /dd/defs ! shell
OS9: chd /d0/lib
OS9: dsave /d0 /dd/lib ! shell
```

The directory /d0/MODULES is not needed unless you keep your system and device module library on your hard disk. I do. Here's how I copied mine:

```
OS9: chd /d0/modules
OS9: dsave -m /d0 /dd/modules ! shell
```

(This assumes that you already have a directory /dd/MODULES and it contains a directory /dd/MODULES/HELP)

The directory /d0/SOURCES is not needed unless you think you would benefit from them.

INSTALLING THE C COMPILER

First thing move the library, etc to the target disk. Place A COPY of the floppy labelled "C Library" in drive /d0. Enter the following commands:

```
OS9: makdir /dd/LIB
OS9: chd /d0/lib
OS9: dsave /d0 /h0/lib ! shell
```

Next move the defs files. Leave the same floppy in and enter:

```
OS9: makdir /dd/DEFS
OS9: chd /d0/defs
OS9: dsave /d0 /h0/defs ! shell
```

Now we need to move the executable Take out the library floppy, and put A COPY of the floppy labelled "C Compiler" into drive /d0. Enter:

```
OS9: chd /dd/cmds
OS9: copy /d0/cmds/cc1 cc1
OS9: copy /d0/cmds/c.prep c.prep
OS9: copy /d0/cmds/c.pass1 c.pass1
OS9: copy /d0/cmds/c.pass2 c.pass2
OS9: copy /d0/cmds/c.opt c.opt
OS9: copy /d0/cmds/c.asm c.asm
OS9: copy /d0/cmds/c.link c.link
```

Now, we need to patch two images, cc1 and c.prep. You can patch them any was you prefer, I prefer to use dEd. These instructions show how to do it with the modpatch utility supplied with OS9 LII.

If you are going to use cc2 or cc then you do not need to patch cc1. Just skip down to the section on patching c.prep. It has to be patched regardless of which front end program you decide to use.

To patch cc1, create and apply the following patch file (illustrated with the build command, use whatever way you prefer to make an ascii text file).

```
OS9: chd /dd/cmds
OS9: build cc1.pat
? 1 cc1
? c EE5 64 44
? c EE6 31 44
? v
? (press CTRL/BREAK)

OS9: load cc1
OS9: rename cc1 old.cc1
OS9: modpatch cc1.pat
OS9: save cc1 cc1
OS9: unlink cc1
OS9: delete old.cc1
```

As noted earlier, you MUST patch c.prep, regardless of which front end program you decide to use. Patch c.prep similarly with the following commands:

```
OS9: chd /dd/cmds
OS9: build c.prep.pat
? 1 c.prep
? c 135C 64 44
'   135D 31 44
? v
? (press CTRL/BREAK)

OS9: load c.prep
OS9: rename c.prep old.c.prep
OS9: modpatch c.prep.pat
OS9: save c.prep c.prep
OS9: unlink c.prep
OS9: delete old.c.prep
```

These patches change the reference to /d1 to /DD in both binaries. You're now ready to do it!! Just change your data directory to the directory which contains your source file(s). Make sure the execution directory is the cmds directory on the target disk.

```
OS9: chx /dd/cmds
OS9: chd /dd/ccwork
```

Assumes you have your C source files in a directory called /dd/ccwork, or in some subdirectory, in which case you should set your data directory all the down to that subdirectory, eg,

```
OS9: chd /dd/ccwork/solitaire
```

To compile a C program whose filename is hello.c you would enter

```
OS9: cc1 hello.c
```

Special instructions:

One way to speed up the compiling process is to pre-load all needed modules into memory first (and presumably, once). This saves on load time when actually running it. Also, it would help to have del and echo already loaded into memory (especially if you are using a floppy system).

```
OS9: chx /dd/cmds
OS9: load cc1
OS9: load c.prep
OS9: load c.pass1
OS9: load c.pass2
OS9: load c.opt
OS9: load c.asm
OS9: load c.link
```

This can only be done on a 512k system, so if you have 128k, don't even bother trying. I personally don't bother pre-loading the system modules, since it only takes a second or two to load the module at runtime from a ST-225 20 meg drive being accessed by a Burke & Burke CoCo-XTC interface.

"Front-End" Program Notes:

CC1 is merely a "front end" to the compiler. It does not actually do any compiling itself, it generates the necessary commands to run each of the 5 steps which are required to perform the compile, writes them to a file called c.com in your current data directory, and forks a process to run that script. Each of the 5 steps reads in a file from the previous step and generates an output file to be read by the next step. Using cc1, all temporary files will reside in the current data direc-tory.

The manual mentions a cc2 needed for Level 2. Forget it, it doesn't exist. Appar-ently when Microware realized that cc1, c.pass1, and c.pass2 would

all work just fine in Level 2, they decided not to develop the promised cc2, c.pass or c.comp.

However, two enterprising software developers have provided us with "replacements" for cc1, both intended to be faster than the original cc1.

The first is called cc2, written by Delphi/Rainbow's own Rick Adams. (He also wrote the CoCo 3 version of Tandy's Shanghai game, nice job, Rick!) It uses the pipe manager to pipe the temporary files from one step to the next. I have used it and it works well. It doesn't need to be patched to make it work with a single disk system. It comes configured to look on drive /DD for all libraries and defs files. It comes with the source, so it is a simple matter of editing the cc2.c file and changing it to suit your needs.

The other one is simply called cc and was written by Carl Kreider. It will use a ramdisk device, /r0, if it is present, to store all temporary files, which makes it the fastest of all, when used with a ramdisk. Note, that a large source module requires a very large ramdisk, which limits what else you can be doing on your system at the same time. In fact you will probably not be able to load all needed modules into memory prior to compiling. Something, as I mentioned earlier, I don't bother with anyway since the load is so fast from a hard drive.

cc uses the defdrive() system call to determine the system device. To feed an appropriate value, an assembler module is supplied with cc which can be modified to suit your system. It can then be compiled and linked with c.asm and c.link, instructions are included in the file ccdevice.a. I have changed mine to point to device /h0 since there is where my LIB and DEFS directories are.

Development Pack Considerations with C:
With the Development Pack you get a new assembler, a new linker and two supplemental libraries, most notable, the Graphics Library documented in the MultiVue Manual. Indeed, to utilize either of these two libraries, you will have to use the new linker provided. This means that you need to tell your "front end" to use the different assembler and linker.

cc1
This is the most difficult. You will need to download CPATCH.AR in the Languages Library on CompuServe. It patches the modules rma (the new assembler) and rlink (the new linker) to think that they are named c.asm and c.link. You will need to delete the original c.asm and c.link prior to installing that patch.

cc2
This is a little easier. Just edit the source file cc2.c and change t' following lines:

```
CHANGE: docmd("C.ASM %s.a -o=%s.r\n",
basename(file), basename(file));

TO READ: docmd("rma %s.a -o=%s.r\n",
basename(file), basename(file));

AND CHANGE: docmd("C.LINK
/dd/lib/cstart.r%s %s -o=%s%s%s\n",

TO READ: docmd("rlink
/dd/lib/cstart.r%s %s -o=%s%s%s\n",
```

Then recompile it with:

```
OS9: cc1 cc2.c
```

cc
This program is also very easy to change, since the source is provided. In fact the instructions for making the change in cc is in the documentation are of the source file itself. Just edit the file cc.h and read t' comments.

Another thing I have done is to merge the libraries sys.l and cgfx.l with clib.l. Oh, before I get to that, I

would also highly recommend downloading "The Kreider Lib". It is a replacement file for clib.l written by Carl Kreider. It fixes problems in the original library plus adds a whole bunch of new calls. Now back to the supplemental libraries. To merge them enter:

```
OS9: chd /dd/lib
OS9: merge cgfx.l sys.l clib.l >n.clib.l
OS9: rename clib.l old.clib.l
OS9: rename n.clib.l clib.l
```

NOTE: THE ORDER OF INPUT FILES IN THE MERGE STATEMENT IS VERY IMPORTANT.. DO NOT CHANGE IT OR YOU WILL GET AN UNUSABLE LIBRARY.

Doing this merge will cause all links to take longer, but you don't need to specify any libraries with the -l option (as depicted on page 10-2 of the M ' iVue manual) anymore. If you make u of the make utility supplied with the Development Pack, you may decide not to merge your libraries in order to be able to minimize your link times.

I hope this little file helps you and speeds you on your way to cranking out some quality C code for the Color Computer 3.

Zack C. Sessions

HOMEWORK by Rodger Alexander

Last month we used Microware's macro editor to create a "startup" file. This month's lesson is an extension of the startup file concept.

Textfiles that are used to actually execute commands or otherwise operate the computer automatically are called "scriptfiles" or "procedure files". The following "scriptfiles" will create a menuing environment for your OS9 system. Use BUILD or EDIT to create your "scriptfiles" or use a text processor.

MAINMENU

```
display 1b 24
list menu.dat
prompt Make Selection:
var.0                    (get selection)
%0                  (execute selection)
```

(NOTE: "prompt" is a basic INPUT type of command utility file included with SHELL+. "var" is a SHELL+ function.)

MENU.DAT

```
echo                  SYSTEM MENU
echo
echo
echo          WP......Word Processor
echo
echo          DC......Dyna Calc
echo
echo          DB......Data Base
echo
echo          TEL.....Telecomm
echo
echo          UT......Utilities
echo
```

(NOTE: With Microware's macro editor or a text processor, it is not necessary to enter "echo" at the beginning of each line. These listing assume you are useing the BUILD utility)

WP

```
echo Place Text Editor Disk in drive 1
echo Press ENTER to continue.
display 1b 24
echo
echo
echo
echo          LOADING TEXT EDITOR
chd /d1
chx /d1/cmds
scred (text editor file to execute)
display 1b 24
chd /d0
chx /d0/cmds
mainmenu
```

DC

```
echo Place Dyna Calc disk in drive 1
echo Press ENTER to continue.
display 1b 24
echo
echo
```

```
echo            LOADING DYNA CALC
chd /d1
chx /d1/cmds
dynacalc </1
display 1b 24
chd /d0
chx /d0/cmds
mainmenu
```

(NOTE: Follow the two above examples for DB, TEL, UT or any other program)

SUMMARY:
In writing the above files, I am assuming that your computer is operating with two disk drives. If you have only one disk drive on your system, change "/d1" to "/d0".

Also, you may want to get a copy of "datamod", a supplimentary SHELL+ program. It will convert the above text files into executable modules that you c n place into your commands direc- /. (Don't forget to ATTRibute them with "e" so they will "execute".)

Better yet, after you "datamod" all of your scriptfiles to the CMDS directory and ATTRibute them for execution, MERGE them together and then LOAD them into memory for instant access/execution.

COMPUTER-PRINTER COMMUNICATIONS

by Scott Honaker

Printers contain a microprocessor, code and character ROM as well as working and buffer RAM. The printer is designed to pass the 128 ASCII characters that are sent to it, to the paper and through special ASCII character sequences, allowing additional characters and commands to be executed.

There are two basic types of computer to printer communications, PARALLEL and SERIAL. A parallel port transmits all 8 data bits simultaneouslv, whereas a A typical dot matrix printer passes all characters from 32-255 (decimal) and prints them. Many characters below ASCII 32 are control strings, the most important are listed below.

6	ACK
7	Bell
8	Backspace
10	Line feed
12	Form feed
13	Carriage return
22	NAK
27	Escape

A serial connection only transmits one bit at a time and has to include svnchronizing start and stop bits. Because of this, a parallel port is typcally much faster than serial, although less reliable over long distance.

In a serial port connection, there is no line that tells the printer how fast the data will be arriving and in what form. Because of this, it is necessary to be sure both the computer and the printer are using the same protacol. The following diagram illustrates how the character 165 would be sent to the printer. Time runs along the horizontal axis and voltage vertically. As baud rate increases, dur duration of each of these pulses is smaller.